Perancangan Perangkat Lunak dengan Metode RSA dengan Pembangkit Bilangan Prima Secara Opsional

Megah Mulya*[1]
[1]Fakultas Ilmu Komputer; Universitas Sriwijaya Palembang, Telp. (0711) 379249
e-mail: ***[1]** megahmulya@yahoo.com

Abstrak

Pengguna dalam memilih algoritma Kriptografi akan juga sangat memperhatikan faktor kecepatan selain kekuatannya. Kompleksitas RSA barkaitan dengan bilangan prima menghasilkan karakteristik kecepatan RSA berbading terbalik dengan kekuatannya. Proses pembangkitan bilangan prima menentukan kekuatan RSA. Jika bilangan yang dibangkitkan berupa prima asli maka menghasilakan penyandian yang kuat. Sedangkan jika yang dibangkitkan berupa bilangan prima yang lemah (pseudo prima) maka menghasilkan penyandian yang lemah pula.

Secara konsep upaya pengamanan data harus sesuai dengan nilai (ekonomis dan strategis) data yang akan diamankan. Data yang bernilai murah cukup diamankan dengan kekuatan yang tidak terlalu besar sedangkan data yang bernilai mahal harus diamankan dengan kekuatan besar Oleh karena itu untuk mengatasi persoalan lambatnya RSA maka dilakukan pemilihan metode pembangkit prima yang sesuai dengan tingkat keamanan yang dibutuhkan pengguna.

Peneitian ini bertujuan untuk melakukan perancangan perangkat lunak yang mengimplementasikan RSA dengan metode pembangkit bilangan prima yang dapat dipilih oleh pengguna. Metode yang digunakan dalam penelitian ini adalah Sieve of Eratosthenes, Sieve of Sundaram dan Sieve of Atkin. Perancangan awal telah dihasilkan dalam penelitian ini berupa diagram use-case, skenario dan kelas analisis. Untuk pengembangan perangkat lunak secara utuh tperlu dilanjutkan ke perancangan detail dan implementsi pada bahasa program tertentu.

Kata kunci: *RSA, Sieve of Eratosthenes, Sieve of Sundaram dan Sieve of Atkin*

1. PENDAHULUAN

Pada saat ini terdapat berbagai macam algoritma Kriptografi simetri maupun asimetri. Penentuan algoritma Kriptografi yang akan digunakan dalam sistem keamanan data selain pertimbangan kekuatan terhadap serangan *Cryptanalis* dan *Brutforse* yang tidak kalah penting adalah pertimbangan kecepatan. Pengukuran kecepatan algoritma kriptografi telah dilakukan oleh berbagai pihak untuk jenis simetri maupun asimetri terutama RSA [1] atau antar varian RSA [2].

Jika suatu algoritma Kriptografi dipercaya kuat namun diketahui lambat dalam proses penyandiannya maka tidak akan dijadikan pilihan oleh pengguna [3]. Pertimbangan kecepatan ini akan menjadi lebih diutamakan lagi jika pemakaian algoritma Kriptografi menyangkut jaringan komputer terutama pada arsitektur *clien-server*. Pada jaringan clien-server dengan jumlah *clien* besar maka lambatnya kinerja algoritma Kriptografi akan sangat signifikan menambah beban keja server. Akumulasi waktu penyadian yang besar akan sangat menjengkelkan pengguna karena dapat menggangu proses lainnya bahkan dapat menimbulkan komputer menggantung (*hang*). Oleh karena itu maka pengguna dalam memilih algoritma Kriptografi akan juga sangat memperhatikan faktor kecepatan selain kekuatannya. Disisi lain

karena kompleksitas RSA barkaitan dengan bilangan prima menghasilkan karakteristik kecepatan RSA berbading terbalik dengan kekuatannya. Proses pembangkitan bilangan prima menentukan kekuatan RSA. Jika bilangan yang dibangkitkan berupa prima asli maka menghasilakan penyandian yang kuat. Sedangkan jika yang dibangkitkan berupa bilangan prima yang lemah (*pseudo prima*) maka menghasilkan penyandian yang lemmah pula.

Secara konsep upaya pengamanan data harus sesuai dengan nilai (ekonomis dan strategis) data yang akan diamankan. Data yang bernilai murah cukup diamankan dengan kekuatan yang tidak terlalu besar sedangkan data yang bernilai mahal harus diamankan dengan kekuatan besar [3]. Oleh karena itu untuk mengatasi persoalan lambatnya RSA maka akan dilakukan pemilihan metode pembangkit prima yang sesuai dengan tingkat keamanan yang dibutuhkan pengguna.

Faktor lambatnya RSA jika dipergunakan dalam pengamanan data tentu harus diimbangi dengan kekuatan pengamanana yang diperoleh. Namun pada kenyataannya tidak setiap proses pengamanan data memerlukan tingkat keamanan yang tinggi. Sedangkan selama ini para pengembang perangkat lunak implementasi RSA pada umumnya memilih metode pembangkit bilangan prima yang paling efektif dengan konsekuensi kompleksitas komputasi tinggi dan diterapkan untuk pengamanan data dengan berbagai macam tingkat kebutuhan pengamanan. Oleh karena itu diperlukan implementasi RSA kedalam perangkat lunak yang memungkinkan pengguna menentukan sendiri metode pembangkit bilangan prima sesuai dengan tingkat keamanan yang dibutuhkan sehingga diharapkan penggunaan RSA menjadi lebih efisien.

Penelitian ini dibatasi oleh hal-hal sebagai berikut: pertama, RSA digunakan secara langsung untuk pengamanan data(pesan). Kedua, data yang diamanakan dengan proses enkripsi dan dekripsi berupa file. Dan yang ketiga, asumsi lingkungan dimana RSA digunakan untuk pengamanan adalah jaringan yang sangat padat lalu-lintas datanya sehingga masalah efisiensi menjadi prioritas.

RSA

Di dalam ilmu Kriptografi RSA merupakan algoritma penyandian berjenis asimetri. Algoritma jenis asimetri mempucai ciri kunci yang digunakan untuk proses enkripsi dan dekripsi berbeda. Skema prose enkripsi dan dekripsi pada sistem Kriptografi asimetri dapat dilihat pada gambar 1.

Gambar 1. Skema Kriptografi asimetri

Pada gambar tersebut terlihat proses enkripsi menggunakan kunci Key 1 dan proses dekripsi menggunakan Key 2.

Algoritma RSA dibuat oleh tiga orang peneliti dari MIT (*Massachussets Institute of Technology*) pada tahun 1976, yaitu: Ron (R)ivest, Adi (S)hamir, dan Leonard (A)dleman. Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima [3]. Terdapat tiga proses utama dalam penggunaan RSA untuk pengamanan data (penyandian) yaitu pembangkitan kunci, enkripsi dan dekripsi. Metode pembangkit bilangan prima menjadi inti dari proses pembangkitan kunci yang menghasilkan sepasang kunci yaitu *public key* dan *private key*.

Kebutuhan adanya sepasang kunci RSA pada saat terdapat dua orang misalnya Alice dan Bob yang ingin berkomunikasi dengan cara mengirim pesan rahasia. Alice ingin Bob mengirimnya sebuah pesan melalui jalur yang tidak aman. Alice akan memberikan *public key* kepada Bob dan menyimpan *private key* untuk dirinya. Langkah-langkah proses pembangkitan sepasang kunci RSA dapat dijelaskan sebagai berikut [3] :

a. Dipilih 2 bilangan prima besar seperti p,q dimana p tidak sama dengan q..
b. Dihitung M = p x q.
c. Dihitung phi(M) = phi(p) * phi(q).
d. Dipilih sebuah integer 'e' dimana 1 < e < phi(M) dan 'e' serta phi(M) adalah *coprime.*.
e. Dihitung 'd' integer sehingga (d * e) mod M = 1.
f. Pasangan nilai (M,e) adalah *public key* dimana M adalah modulo dan e adalah eksponen encryption.
g. Pasangan nilai (M,d) adalah *private key* dimana M adalah modulo dan d adalah eksponen decryption.

Pemfaktoran dilakukan untuk memperoleh kunci privat. Selama pemfaktoran bilangan besar menjadi faktor-faktor prima belum ditemukan algoritma yang mangkus, maka selama itu pula keamanan algoritma RSA tetap terjamin. RSA sampai sat ini masih dinyatakan kuat terutama bila menggunakan kunci dengan panjang minimal 512 bit [4]. Kekuatan RSA yang bertumpu pada sulitnya pemfaktoran bilangan besar menjadi faktor prima menyebabkan kompleksitas yang tinggi dalam komputasinya. Hal itu menimbulkan konsekuensi performansi RSA sangat lambat jika dimabandingkan dengan algoritma kriptografi lain yang berjenis simetri hingga seratus kali lebih lambat [3]. Beberapa varian RSA telah dihasilkan untuk berbagai keperluan namun semuanya masih tergolong lambat [4].

Pembangkit Bilangan Prima

Pembangkitan bilangan prima sangat penting dalam Kriptografi terutama karena RSA menggunakan pembangkit bilangan prima untuk mendapatkan kunci penyandian. Terdapat berbagai metode pembangkit bilangan prima diantaranya adalah Sieve of Eratosthenes, Sieve of Sundaram dan Sieve of Atkin dan lain sebagainya [5]. Selain metode–metode tersebut sampai saat ini masih dilakukan penelitian penggunaan metode-metode lain untuk pembangkitan bilanmgan prima. Metode-metode tersebut memiliki kelebihan dan kelemahannya masing-masing

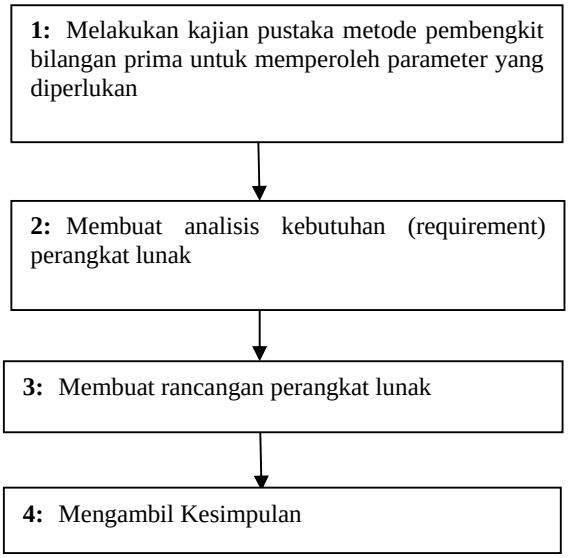
Banyak penelitian dilakukan untuk mendapatkan metode yang paling cepat dalam pembangkitan bilangan prima. Penelitian tentang pembangkit bilangan prima dengan komputer paralel dilakukan oleh Hwang Soonwook dan Kyusik Chung pada tahun 2006 dengan metode *Sieve of Eratosthenes.* Hasil penelitian tersebut menunjukkan pada saat itu untuk mencapai kecepatan yang dianggap ideal pembangkitan bilangan prima tidak dapat dilakukan oleh proses tunggal dalam RSA [6].

Penelitian Zhang Qing dan Hu Zhihua pada tahun 2011 menghasilkan kesimpulan bahwa upaya mendapatkan kekuatan RSA untuk membangkitkan bilangan prima dengan algoritma genetik terbatasi pada kenyataan ukuran bilangan prima yang besar yang pemfaktorannya akan lambat [4].

Penelitian Dongjiang, Li dan Wang Yandan pada tahun 2012 dengan tujuan mengurangi waktu dan sumberdaya memori yang digunakan pada pembangkitan kunci RSA. Pada penelitian tersebut terdapat ide berupa pra penyaringan prima baru kemudian dilanjutkan dengan algoritma Miller-Rabin untuk mendapatkan nilai akhir prima dan hasilnya peningkatan proses pembangkitan kunci RSA [7].

2. METODE PENELITIAN

Penelitian ini dilakukan di laboratorium dan di lapangan dengan skema seperti terlihat pada gambar 2.

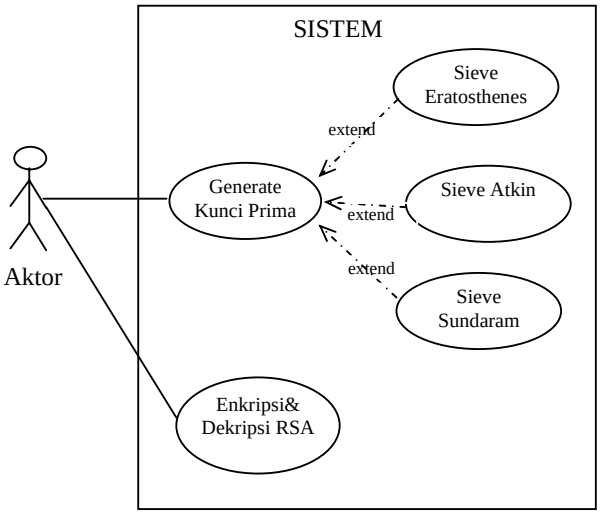


Gambar 2. Skema langkah-langkah penelitian

3. HASIL DAN PEMBAHASAN

Untuk mendapatkan perancangan terlebih dahulu dilakukan analisisHasil analisis dan perancangan berupa pemodelan statis dituangkan dalam bentuk diagram-diagram UML yang disesuaikan dengan tahap-tahap pada RUP. Hasil tersebut berupa diagram use-case, skenario dan diagram kelas analisis.

Use-case diagram



Gambar 3. Skema langkah-langkah penelitian

Skenario

Tabel 1. Skenario use-case generate kunci

Nama *Use Case*	:	Generate Kunci Prima
Aktor	:	Pengguna
Tujuan	:	Membangkitkan kunci bilangan prima
Deskripsi	:	*Use case* ini digunakan untuk membangkitkan kunci bilangan prima sebagai kunci RSA.
Kondisi Awal	:	Tidak ada
Kondisi Akhir	:	Terbangkitkan kunci bilangan prima RSA

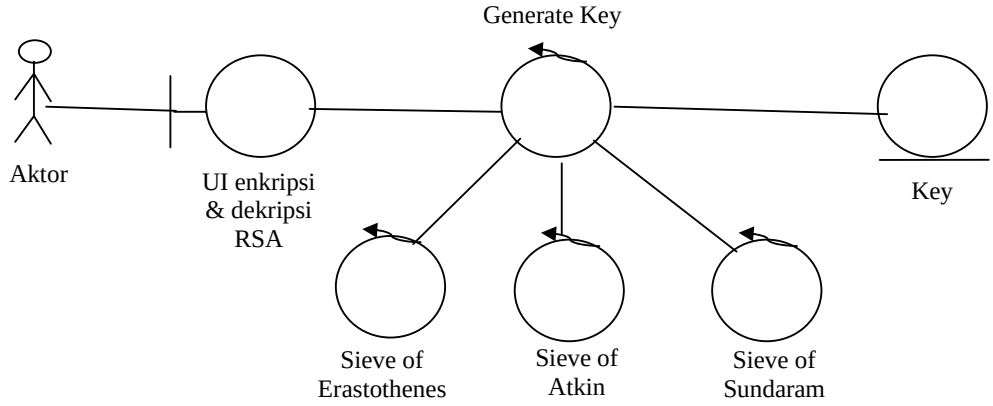
Aktor	Sistem
1. Memilih menu generate kunci RSA	
	2. Menampilkan form menu pilihan metode
3. Memilih metode pembangkitan	
	4. Melakukan pmbangkitan bilangan prima
	5. Melakukan pembangkitan kunci RSA
	6. Menyimpan kunci RSA kedalam media
	7. Menampilkan pesan pembangkitan kunci telah selesai
8. Menklik tombol OK	

Tabel 2. Skenario use-case enkripsi & dekripsi RSA

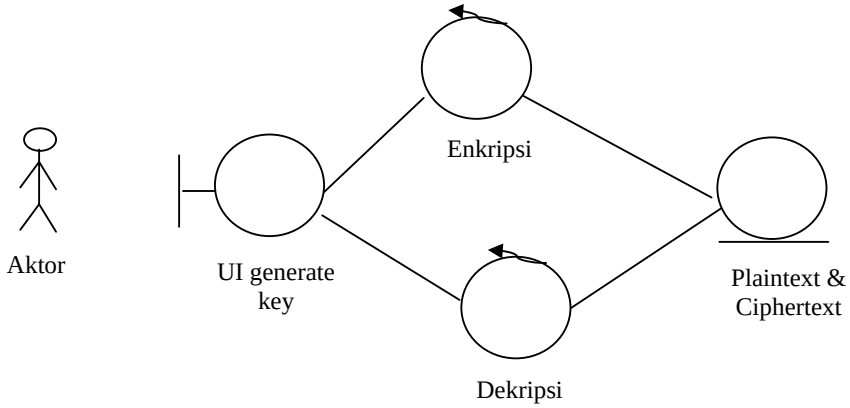
Nama *Use Case*	:	Enkripsi & Dekripsi RSA
Aktor	:	Pengguna
Tujuan	:	Melkukan pengamanan pesan dengan RSA
Deskripsi	:	*Use case* ini digunakan untuk mengamankan pesan melalui proses enkripsi dan dekripsi.
Kondisi Awal	:	Kunci RSA telah terbangkitkan
Kondisi Akhir	:	Pesan telah dienkripsi RSA dan siap juga untuk didekripsi

Aktor	Sistem
1. Memilih menu enkripsi & dekripsi	
	2. Menampilkan form menu pilihan enkripsi atau
3. Memilih file yang berisi data yang	
	4. Melakukan enkripsi/dekripsi terhadap file yang
	5. Menampilkan informasi permintaan file yang digunakan untuk menyimpan hasil
6. Memilih/meginputkan nama file untuk	
	7. Menampilkan pesan proses enkripsi/dekripsi telah selesai
8. Mengklik tombol OK	

Diagram kelas analisis



Gambar 4. Diagram kelas analisis use-case generate kunci.



Gambar 5. Diagram kelas analisis use-case enkripsi & dekripsi RSA

4. KESIMPULAN

Perangkat lunak implementasi RSA dengan pemangkit bilangan prima secara opsional telah berhasil dirancang. Dari hasil rancangan diketahui perangkat lunak memiliki dua fitur fungsional yaitu generate prima dan enkripsi&dekripsi RSA. Skenario kedua fitur fungsional tersebut juga telah dihasilkan. Fitur fungsional yang pertama tersusun oleh enam kelas (campuran antara 1 bondery, 4 controler dan 1 entitas). Fitur fungsional kedua tersusun oleh empat kelas (campuran 1 bondery, 2 controler dan 1 entitas)

5. SARAN

Penelitian yang telah dihasilkan ini dapat digunakan sebagai rancangan sehingga dapat diimplementasikan dalam sebuah sistem ataupun perangkat lunak.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada semua pihak yang telah memberi dukungan sehingga terbitnya jurnal ini, terimakasih besar kepada redaksi jurnal Jupiter yang telah berkenan menerbitkan naskah jurnal ini.

DAFTAR PUSTAKA

[1] Vivek, B.Kute, P. R. Paradhi and G. R. Bamnote, 2009,. A Software Comparison Of RSA an ECC., *Intenational Journal of Computer Science and Aplications*

[2] Mamun, A., Mohammad M.I., S.M. Mashihure R.A.H and Salah Uddin Ahmad, 2008, Performance Evaluation of Several Efficient RSA Variants, *(IJCSNS) International Journal of Computer Science and Network Security, VOL.8 No.7, July 2008*

[3] Schneier B., 1999, *Applied Cryptography*, John Wiley & Sons

[4] Qing, Zhang and Hu Zhihua, 2011, The large prime numbers generation of RSA algorithm based on genetic algorithm , *International Conference on Intelligence Science and Information Engineering*

[5] Ribenboim, Paulo, 1989, *The Book of Prime Number Records*, Springer- Verlag.

[6] Hwang Soonwook and Kyusik Chung ,2006, Load Balanced Parallel Prime Number Generator with Sieve of Eratosthenes on Cluster Computers, *Seventh International Conference on Computer and Information Technology.*

[7] Dongjiang, Li and Wang Yandan, 2012, The research on key generation in RSA public-key cryptosystem, *Fourth International Conference on Computational and Information Sciences*